Algorithm Generate recommend path

```
Input :  $s$, $D$, $T$, $l$
  $s$ : Source attraction
  $D$ : $\{\text{attraction}_i \mid \text{attraction}_i \in \text{same region of } S\}, \forall \text{attraction}_i \in \text{database}$
  $T$ : The limitation of driving time for each 2 attractions
  $l$ : Required number of tourist attraction in the recommend path

Output :  recommend path, total driving time, $Labels$

Initialize :
  cur := $s$
  $Labels := s.Label$
  $drivingtime := 0$
  $Path := \{s\}$
1: while $|Path| < l$ do
2:     target := $\emptyset$
3:     Candidate $\leftarrow \{c_j \mid \text{d(cur, } c_j) \le T\}, \forall c_j \in D, c_j \neq \text{cur}$
4:     if Candidate is empty then
5:         return NoCandidate
6:     end if
7:
8:     $n_{max} \leftarrow$ max(#nonzero entry of $(Labels + c_i.Label)$),$\forall c_i \in$ Candidate
9:     Targets $\leftarrow \mathbf{argmax}_{c_i}$(#nonzero entry of $(Labels + c_i.Label)$ )
10:
11:    if $n_{max}$ doesn't improve then
12:        target $\leftarrow \mathbf{argmin}_{c_i}(\mathbf{Variance}((Labels + c_i.Label))+\text{d(cur, } c_i))$
13:        ,$\forall c_i \in$ Candidate
14:    else
15:        if |targets| == 1 then
16:            target $\leftarrow$ targets[0]
17:        else if |targets| > 1 then
18:            target $\leftarrow \mathbf{argmin}_{t_i}(\mathbf{Variance}((Labels + t_i.Label))+\text{d(cur, } t_i))$
19:            ,$\forall t_i \in$ Targets
20:        end if
21:    end if
22:
23:    $Path \leftarrow Path \cup$ target
24:    $drvingtime \leftarrow$ d(cur, target)
25:    $Labels \leftarrow Labels$+target.$Label$
26:    cur $\leftarrow$ target
27: end while
28: return $Path, Labels, drivingtime$
```